

## **Project 1**

### **Building a CI/CD Pipeline for a Retail Company**

First of all, thanks very much to edureka and instructors for giving me confidence that I can do any type of devops projects. I just loved working on this project. I have chosen project 1 out of two projects. So I implemented the 1st project.

github repository: <https://github.com/bejoykoottumkal/devops-lab.git>

### **Business Challenge/Requirement:**

ABC Technologies is a leading online retail store, and it has recently acquired a large retail offline business store. The business store has a large number of stores across the globe but is following the conventional pattern of development and deployment. As a result, it has landed at a great loss and is facing the following challenges.

- Low available
- Low scalable
- Low performance
- Hard to build and maintain
- Developing and deploying are time-consuming

ABC will acquire the data from all these storage systems and plans to use it for analytics and prediction of the firm's growth and sales prospects. In the first phase, ABC has to create the servlets to add a product and display product details. Add servlet dependencies required to compile the servlets. Create an HTML page that will be used to add a product. The team is using Git to keep all the source code.

ABC has decided to use the DevOps model. Once source code is available in GitHub, we need to integrate it with Jenkins and provide continuous build generation for continuous delivery as well as integrate with Ansible and Kubernetes for deployment. Use Docker Hub to pull and push images between Ansible and Kubernetes.

### **Problem Statements/Tasks:**

We need to develop a CI/CD pipeline to automate the software development, testing, packaging, and deployment, reducing the time to market the app and ensuring good quality service is experienced by end users. In this project, we need to—

- Push the code to our GitHub repository.

- Create a continuous integration pipeline using Jenkins to compile, test, and package the code present in GitHub.
- Write Dockerfile to push the war file to the Tomcat server.
- Integrate Docker with Ansible and write the playbook.
- Deploy artifacts to the Kubernetes cluster
- Monitor resources using Prometheus.

## **Approach to Solve:**

**Task 1:** Clone the project from the GitHub link shared in resources to your local machine. Build the code using Maven commands.

**Task 2:** Set up the Git repository and push the source code. Then, log in to Jenkins.

1. Create a build pipeline containing a job for each
  - One for compiling source code
  - Second for testing source code
  - Third for packing the code
2. Execute the CI/CD pipeline to execute the jobs created in step 1
3. Set up a master-slave node to distribute the tasks in the pipeline

**Task 3:** Write a Docket file. Create an Image and container on the Docker host. Integrate docker host with Jenkins. Create CI/CD job on Jenkins to build and deploy on a container.

1. Enhance the package job created in step 1 of task 2 to create a docker image.
2. In the Docker image, add code to move the war file to the Tomcat server and build the image.

**Task 4:** Integrate the Docker host with Ansible. Write an Ansible playbook to create an image and create a continuer. Integrate Ansible with Jenkins. Deploy Ansible-playbook. CI/CD job to build code on ansible and deploy it on docker container

- Deploy Artifacts on Kubernetes
- Write pod, service, and deployment manifest file
- Integrate Kubernetes with Ansible
- Ansible playbook to create deployment and service

**Task 5:** Using Prometheus, monitor the resources like CPU utilisation: Total Usage, Usage per core, usage breakdown, memory, and network on the instance by providing the endpoints on the local host. Install the node exporter and add the URL to the target in Prometheus.

-----

## **Let's start...**

### **Task 1:**

**Clone the project from the GitHub link shared in resources to your local machine. Build the code using Maven commands.**

#### **Approach i have followed:**

Before setting up the CI/CD pipeline,

I checked and installed necessary software in my machine using power-shell and also installed Ubuntu console using the following codes.

1. Java:

Verified Java Installation:

```
java -version
```

2. Maven:

Verified Maven Installation:

```
mvn -version
```

3. Git:

Verified Git Installation:

```
git --version
```

4. Jenkins:

Verified Jenkins Installation:

Accessed Jenkins in a web browser at <http://localhost:8080> (default) and follow the setup process.

5. Docker:

Verified Docker Installation:

```
docker –version
```

6. Kubernetes:

Verified Kubernetes Installation:

kube CTL

in AWS t3 Medium

## 7. Ansible:

Verify Ansible Installation:

```
ansible --version
```

in AWS t3 Medium

## 8. Prometheus:

Verified Prometheus Installation:

Access Prometheus in a web browser at <http://localhost:9090>.

### **Additional Considerations :**

- Ensured Service Availability
- Checked Jenkins, Docker, Ansible, Kubernetes , and Prometheus services are running and available.

### **Configuration Check:**

- Checked and Verified the configurations for Jenkins, Docker, Ansible, Kubernetes and Prometheus.
- Network Ports
- Ensured that required ports are not blocked by firewalls. For example, Jenkins typically uses port 8080 and Prometheus uses port 9090.
- Documentation:
- Record Versions:
- Documented the versions of each installed software.
- As a given project is based on Java I have used Maven as a build tool to build the code. Maven and Java are installed already in the master machine. I have run the Maven compile, test, and package tasks successfully in the local environment and produced the results for each and also executed the Maven clean install command to build the code.
- Before this I have downloaded the sample Java code from the LMS into my own local environment and used git bash to upload to the below newly created public GitHub repository.

**Repository:** <https://github.com/bejoykoottumkal/devops-lab.git>

## Step 1: Cloned the Project

Opened a terminal or command prompt on your local machine and ran the following command to clone the project from the GitHub repository:

```
# git clone https://github.com/bejoykoottumkal/devops-lab.git
```

## Step 2: Navigated to the Project Directory

Moved to the project directory using the cd command:

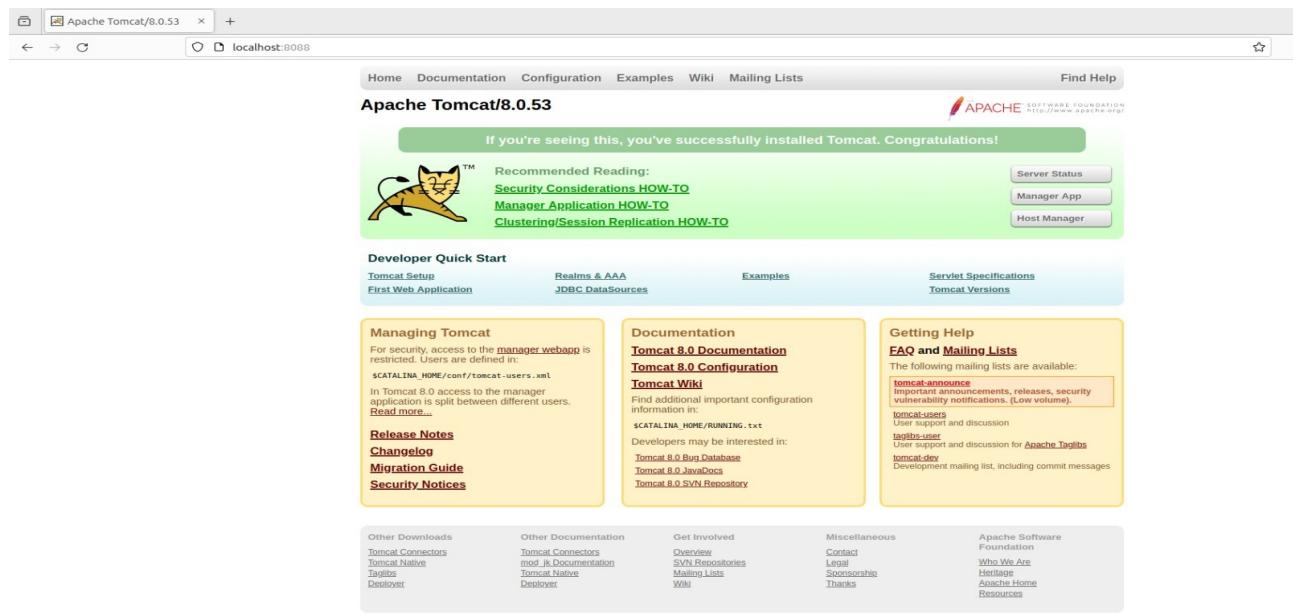
```
# cd devops-lab
```

## Step 3: Built the Code with Maven

Used Maven to build the project. Executed the following commands:

```
# mvn clean  
# mvn compile  
# mvn test  
# mvn package
```

## Step 4: Application Deployment using native tomcat installation for checking the whether application is up.



**Copied the war file created in target folder to the webapps folder of tomcat and stared service. Application was available with url : <http://localhost:8088/cicd/>**



## References:

<https://www.marcobehler.com/guides/mvn-clean-install-a-short-guide-to-maven>

<https://www.geeksforgeeks.org/maven-lifecycle-and-basic-maven-commands/>

## Task 2:

**Set up the Git repository and push the source code. Then, log in to Jenkins.**

**1. Create a build pipeline containing a job for each**

- One for compiling source code
- Second for testing source code
- Third for packing the code

**2. Execute the CI/CD pipeline to execute the jobs created in step 1**

**3. Set up a master-slave node to distribute the tasks in the pipeline**

## Approach i have followed:

### **Step 1: Installed Jenkins**

Downloaded and installed Jenkins on the machine, following the official instructions:

<https://www.jenkins.io/download/>

## Started Jenkins after installation and log in



## Sign in to Jenkins

Username

Password

Keep me signed in

**Sign in**

## Global Tool Configurations.

Dashboard > Manage Jenkins

**Manage Jenkins**

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

**Manage Jenkins**

My Views

Open Blue Ocean

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#).

**Set up agent** **Set up cloud** **Dismiss**

Warnings have been published for the following currently installed components:

**Build Pipeline Plugin 2.0.1:**  
Stored XSS vulnerability (no fix available)  
No fixes for these issues are available. It is recommended that you review the security advisory and apply mitigations if possible, or uninstall this plugin.

[Go to plugin manager](#) [Configure which of these warnings are shown](#)

Restore the previous version of Jenkins [Downgrade to 2.387.3](#)

**System Configuration**

No builds in the queue.

**System** Configure global settings and paths

**Tools** Configure tools, their locations and

**Plugins** 35 Add, remove, disable or enable plugins

## Added JAVA HOME.

JDK installations

JDK installations ^

Add JDK

**JDK**

Name

JAVA\_HOME

Install automatically ?

Add Installer ▾

## Added MAVEN HOME

The screenshot shows the 'Maven installations' configuration screen. A new Maven installation is being added with the name 'M2\_HOME' and the path 'B:\apache-maven-3.9.6\'. The 'Install automatically' checkbox is checked, and the 'Version' dropdown is set to '3.9.6'. There are 'Save' and 'Apply' buttons at the bottom.

## Step 2: Created Jenkins Build Pipeline

Opened Jenkins in the web browser (<http://localhost:8080>).

Installed the necessary plugins.

The screenshot shows the Jenkins plugin manager. Under the 'Installed plugins' tab, several plugins are listed, all of which are currently enabled (indicated by a green switch icon). The listed plugins include Ansible plugin, Ant Plugin, Apache HttpComponents Client 4.x API Plugin, Apache HttpComponents Client 5.x API Plugin, and Authentication Tokens API Plugin.

Name	Enabled
Ansible plugin	Enabled
Ant Plugin	Enabled
Apache HttpComponents Client 4.x API Plugin	Enabled
Apache HttpComponents Client 5.x API Plugin	Enabled
Authentication Tokens API Plugin	Enabled

## Step 3: Configured Jenkins Jobs

Created a new Jenkins job.

Configured it to pull the code from the Git repository.

The screenshot shows the 'Configure' screen for a Jenkins job. The 'Pipeline' tab is selected. Under the 'SCM' section, 'Git' is chosen. The 'Repository URL' is set to `https://github.com/bejoykoottumkal/devops-lab.git`. The 'Credentials' dropdown is set to '- none -'. There is an 'Advanced' button and a 'Save' button at the bottom.

## Step 4: Created a Build Pipeline

Created a new Jenkins job of type "Pipeline" added the pipeline script in it.

The screenshot shows the 'Configure' screen for a Jenkins pipeline job. The 'Pipeline' tab is selected. The 'Definition' dropdown is set to 'Pipeline script'. The 'Script' editor contains the following Jenkinsfile:

```
1* pipeline {
2  agent any
3
4  // Define variables
5  environment {
6    GIT_REPO = "https://github.com/bejoykoottumkal/devops-lab.git"
7    // dckr_dat is the id used when defining the Docker Hub credentials in Jenkins.
8    // DOCKER_HUB_CREDENTIALS=credentials('dckr_bejoy')
9    DOCKER_HUB_REPO = "bejoykoottumkal/devops-lab"
10   ANSIBLE_SERVER_IP="52.99.96.184"
11 }
12
13 tools {
14   maven "maven-3.6.3"
15   git "Default"
16 }
17
18 stages {
19
20   stage('Stage 1 - Checkout Code') {
21     steps {
22       script {
23         properties([pipelineTriggers([pollSCM('')])])
24         //Get the code form GITHUB
25         git GIT_REPO
26       }
27     }
28   }
29 }
30
31 stage('Stage 2 - Compile Code') {
32   steps {
33     //cmd to compile the code
34   }
35 }
```

The 'Save' button is highlighted.

## JenkinFile

```
39      stage('Stage 3 - Run Unit Tests') {
40          steps {
41              //cmd to run tests
42              sh "mvn test"
43          }
44      }
45
46      stage('Stage 4 -Create package') {
47          steps {
48              //cmd to create the build of project
49              sh "mvn package"
50          }
51      }
52
53      stage('Stage 5 - clean install') {
54          steps {
55              //cmd to create the build of project
56              sh "mvn clean install"
57          }
58      }
```

## Target folder in Jenkins:

The screenshot shows the Jenkins workspace browser interface. The URL in the address bar is `localhost:8080/job/CI_PIPELINE/140/execution/node/3/ws`. The page title is "Jenkins". The breadcrumb navigation shows: Dashboard > CI\_PIPELINE > #140 > Allocate node:Start > Workspace > Workspace. On the left, there are links for Status, Console Output, and Workspace (which is currently selected). The main area is titled "Workspace" and shows a file tree. The tree includes .git, cicd, docs, server, and webapp. Under webapp, there are four files: deployment.yml, hosts, pom.xml, and README.md. Each file has a timestamp, size, and two small icons next to them.

File	Last Modified	Size	Actions
deployment.yml	Jan 13, 2024, 6:49:05 PM	686 B	🔗 📺
hosts	Jan 13, 2024, 3:20:45 PM	27 B	🔗 📺
pom.xml	Jan 14, 2024, 4:13:44 PM	6.18 KiB	🔗 📺
README.md	Jan 16, 2024, 4:14:52 PM	20 B	🔗 📺

[\(all files in zip\)](#)

## Execute the CI/CD pipeline to execute the jobs created in step 1 CI\_PIPELINE:

The screenshot shows the Jenkins interface for the CI\_PIPELINE job. On the left, there's a sidebar with options like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, Pipeline Syntax, and Git Polling Log. The main area is titled "Stage View" and displays a grid of execution times for different stages across three builds. The columns represent stages: Declarative: Tool Install, Stage 1 - Checkout Code, Stage 2 - Compile Code, Stage 3 - Run Unit Tests, Stage 4 - Create package, Stage 5 - clean install, Build Docker Image, Login to Docker Hub, Push Image to Docker Hub, Deploy to K8s, and Declarative: Post Actions. The rows show build details: #137 (Jan 18, 2024, 1:25 PM), #136 (Jan 18, 2024, 11:28 AM), and #135 (Jan 18, 10:56). Average stage times are listed as 120ms, 1s, 1s, 1s, 1s, 2s, 3s, 3s, 8s, 10s, and 48ms respectively.

	Declarative: Tool Install	Stage 1 - Checkout Code	Stage 2 - Compile Code	Stage 3 - Run Unit Tests	Stage 4 - Create package	Stage 5 - clean install	Build Docker Image	Login to Docker Hub	Push Image to Docker Hub	Deploy to K8s	Declarative: Post Actions
Average stage times: (Average full run time: ~56s)	120ms	1s	1s	1s	1s	2s	3s	3s	8s	10s	48ms
#137	139ms	1s	1s	1s	1s	2s	2s	2s	7s	31s	
#136	125ms	943ms	1s	1s	1s	2s	2s	2s	12s	28s	
#135	123ms	924ms	921ms	1s	1s	2s	1s	2s	8s	1s	44ms

And I have configured the git polling as a cron job for a compile job that triggers for github code so that if whatever change happens then it will trigger and perform the given task.

## Poll SCM settings using cron job in compile job(triggers every 5 minutes):

The screenshot shows the Jenkins configuration page for the CI\_PIPELINE job. Under the "Configuration" tab, the "Build Triggers" section is expanded. It includes options for General, Advanced Project Options, and Pipeline. The "Poll SCM" option is checked, and its schedule is set to "H/5 \* \* \* \*". A note below says "No schedules so will only run due to SCM changes if triggered by a post-commit hook". There are also checkboxes for "Ignore post-commit hooks" and "Quiet period". At the bottom are "Save" and "Apply" buttons.

## Console output

Dashboard > CI\_PIPELINE > #128

```
PLAY RECAP -----
54.145.170.236 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 39969 killed;
[ssh-agent] Stopped.
[Pipeline] // sshagent
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

## Pipeline Steps

Jenkins

localhost:8080/job/CI\_PIPELINE/139/flowGraphTable/

Search (CTRL+K)

Dashboard > CI\_PIPELINE > #139 > Pipeline Steps

Step	Arguments
Start of Pipeline - (20 sec in block)	
node - (20 sec in block)	
node block - (20 sec in block)	
withCredentials - (20 sec in block)	
withCredentials block - (20 sec in block)	
withEnv - (20 sec in block)	GIT_REPO, DOCKER_HUB_REPO, ANSIBLE_SERVER_IP
withEnv block - (20 sec in block)	
stage - (0.1 sec in block)	Declarative: Tool Install
stage block (Declarative: Tool Install) - (77 ms in block)	
tool - (14 ms in self)	maven-3.6.3
envVarsForTool - (20 ms in self)	
tool - (14 ms in self)	Default
envVarsForTool - (13 ms in self)	
withEnv - (20 sec in block)	M2_HOME, MAVEN_HOME, PATH+MAVEN
withEnv block - (20 sec in block)	
stage - (0.95 sec in block)	Stage 1 - Checkout Code

## Node Details.

The screenshot shows the Jenkins 'Nodes' page. On the left, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (1 idle, 2 idle). The main area is titled 'Nodes' and contains a table with one row. The table columns are: S, Name, Architecture, Clock Difference, Free Disk Space, Free Swap Space, Free Temp Space, and Response Time. The single node listed is 'Data obtained', which is a 'Built-In Node' running 'Linux (amd64)'. It is 'In sync' with a clock difference of '6 min 57 sec'. It has 254.37 GiB free disk space, 976.00 MiB free swap space, and 254.37 GiB free temp space. Its response time is 0ms.

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Data obtained	Built-In Node Linux (amd64)	In sync 6 min 57 sec	254.37 GiB	976.00 MiB	254.37 GiB	0ms

## Node Build Details.

The screenshot shows the Jenkins 'Allocate node : Start' page. The left sidebar has links for 'Up', 'Status' (selected), 'Console Output', and 'Workspace'. The main content area shows a green checkmark icon and the text 'Node Allocate node : Start'. Below it is a 'Parents' section with a single item: 'Start of Pipeline'.

## References:

<https://www.jenkins.io/blog/2022/06/28/require-java-11/>

<https://www.jenkins.io/doc/administration/requirements/upgrade-java-guidelines/#:~:text=If%20you're%20upgrading%20your,screen%20of%20your%20Jenkins%20instance.>

<https://stackoverflow.com/questions/69495517/unable-to-install-jenkins-on-ubuntu-20-04>

<https://stackoverflow.com/questions/14119983/java-home-and-path-are-set-but-java-versionstill-shows-the-old-one>

## Task 3:

**Write a Docket file. Create an Image and container on the Docker host. Integrate docker host with Jenkins. Create CI/CD job on Jenkins to build and deploy on a container.**

1. Enhance the package job created in step 1 of task 2 to create a docker image.
2. In the Docker image, add code to move the war file to the Tomcat server and build the image

## Approach i have followed:

### **Step 1: Created a Docker file**

Created a Docker file in the root directory. The file has the instructions for building Docker image.

#### **Docker file**



```
Code Blame 4 lines (3 loc) · 130 Bytes ⚡ Code 55% faster with GitHub Copilot  
1 FROM tomcat:latest  
2 RUN cp -R /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps  
3 COPY ./*.war /usr/local/tomcat/webapps
```

This Docker file installed Docker inside the Jenkins image and copied Jenkins job configurations from the jobs directory to the appropriate location inside the container.

### **Step 2: Built the Docker Image**

Built the Docker image using the following command within the Jenkinsfile:

```
60      stage('Build Docker Image') {  
61          steps {  
62              sh 'sudo docker build -t ${DOCKER_HUB_REPO}:${BUILD_NUMBER} .'  
63              echo 'Build Image Completed'  
64          }  
65      }  
66
```

### **Step 3: Ran the Docker Container**

Ran the Docker container :

```
1 ---  
2 - hosts: ansible-server  
3   become: true  
4  
5   tasks:  
6  
7     - name: Login to Dockerhub  
8       command: sudo docker login -u bejoykoottumkal@gmail.com -p dckr_pat_KKj6tEX1WMiDnqPcA3oIJMd7kos  
9  
10    - name: create docker image using war file  
11      command: sudo docker build -t bejoykoottumkal/devops-lab:latest .  
12      args:  
13        chdir: /opt/k8s-lab  
14  
15    - name: create tag to image  
16      command: sudo docker tag bejoykoottumkal/devops-lab bejoykoottumkal/devops-lab  
17  
18    - name: push image on to dockerhub  
19      command: sudo docker push bejoykoottumkal/devops-lab:latest
```

Mapped the Docker socket into the container, allowing Jenkins to communicate with the Docker daemon on the host.

### **Step 4: Integrated Docker Host with Jenkins**

In the Jenkins dashboard, navigated to "Manage Jenkins" -> "Manage Plugins" -> "Available."

Searched for and installed the "Docker" plugin.

**Plugins**

Updates

35

Available plugins

Installed plugins

Advanced settings

Docker

/

Install



Install Name ↓

Released

 Docker 1.5

Cloud Providers Cluster Management docker

4 mo 19 days ago

This plugin integrates Jenkins with Docker

 Docker Commons 439.va\_3cb\_0a\_6a\_fb\_29

Library plugins (for use by other plugins) docker

6 mo 15 days ago

Provides the common shared functionality for various Docker-related plugins.

 Docker Pipeline 572.v950f58993843

pipeline DevOps Deployment docker

5 mo 13 days ago

Build and use Docker containers from pipelines.

 Docker API 3.3.4-86.v39b\_a\_5ede342c

Library plugins (for use by other plugins) docker

1 mo 24 days ago

This plugin provides docker-java API for other plugins.

This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

**Docker Installation Complete****Plugins**

Updates

35

Available plugins

Installed plugins

Advanced settings

Download progress

**Download progress**

## Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

Cloud Statistics

Success

Docker Commons

Success

Apache HttpComponents Client 5.x API

Success

Docker API

Success

Docker

Success

Loading plugin extensions

Success

[Go back to the top page](#)

(you can start using the installed plugins right away)

 [Restart Jenkins when installation is complete and no jobs are running](#)

## Step 5: Created an account in the Docker Hub

Created a repository : bejoykoottumkal/devops-lab

The screenshot shows the Docker Hub interface for the repository 'bejoykoottumkal/devops-lab'. At the top, there's a navigation bar with 'docker hub', 'Explore', 'Repositories' (which is selected), and 'Organizations'. A search bar says 'Search Docker Hub'. Below the navigation, it shows 'bejoykoottumkal / Repositories / devops-lab / General'. A message says 'Add a short description for this repository' with a note: 'The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results.' There's an 'Update' button. The main card for 'bejoykoottumkal / devops-lab' shows a 'Docker commands' section with 'To push a new tag to this repository:' and the command 'docker push bejoykoottumkal/devops-lab:tagname'. Below this is a 'Public View' button. The 'Description' section notes 'This repository does not have a description'. The 'Last pushed' time is '4 days ago'. On the left, a 'Tags' section lists several tags: 140, 139, 138, latest, and 137, all of which are 'Image' type tags. On the right, an 'Automated Builds' section is available with a 'Upgrade' button. At the bottom, there's a 'Repository overview' link.

## Step 6: Login to Docker HUB

Using the pipeline script created a stage for login to docker hub account

```
stage('Login to Docker Hub') {
    steps{
        sh 'echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
        echo 'Login Completed'
    }
}
```

## Step 7: Pushed the image to the Docker Hub

```
stage('Push Image to Docker Hub') {
    steps {
        sh 'sudo docker push ${DOCKER_HUB_REPO}:$BUILD_NUMBER'
        echo 'Push Image Completed'
    }
}
```

## **References:**

<https://plugins.jenkins.io/role-strategy/>

<https://stackoverflow.com/questions/37603621/jenkins-sudo-no-tty-present-and-no-askpass#program-specified-with-nopasswd>

<https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.68/bin/apache-tomcat-9.0.68.tar.gz>

<https://phoenixnap.com/kb/how-to-configure-docker-in-jenkins>

[https://www.tutorialspoint.com/docker/docker\\_continuous\\_integration.htm](https://www.tutorialspoint.com/docker/docker_continuous_integration.htm)

<https://www.provartesting.com/documentation/devops/continuous-integration/docker/setting#up-continuous-integration-with-jenkins-for-docker/>

<https://www.youtube.com/watch?v=mszE-OCI2V4&list=PLVz2XdJiJQxwS0BZUHX34ocLTJtRGSQzN&index=5>

<https://www.digitalocean.com/community/questions/how-to-fix-docker-got-permission-denied-while-trying-to-connect-to-the-docker-daemon-socket>

<https://stackoverflow.com/questions/50798720/jenkins-throwing-error-jenkins-model-invalidbuildsdir-item-rootdir-builds-d>

<https://linuxize.com/post/how-to-list-groups-in-linux/>

<https://stackoverflow.com/questions/17733671/how-can-i-tell-what-user-jenkins-is-running-as>

<https://stackoverflow.com/questions/55156958/jenkins-fail-to-deploy-war-to-tomcat-container-second-time>

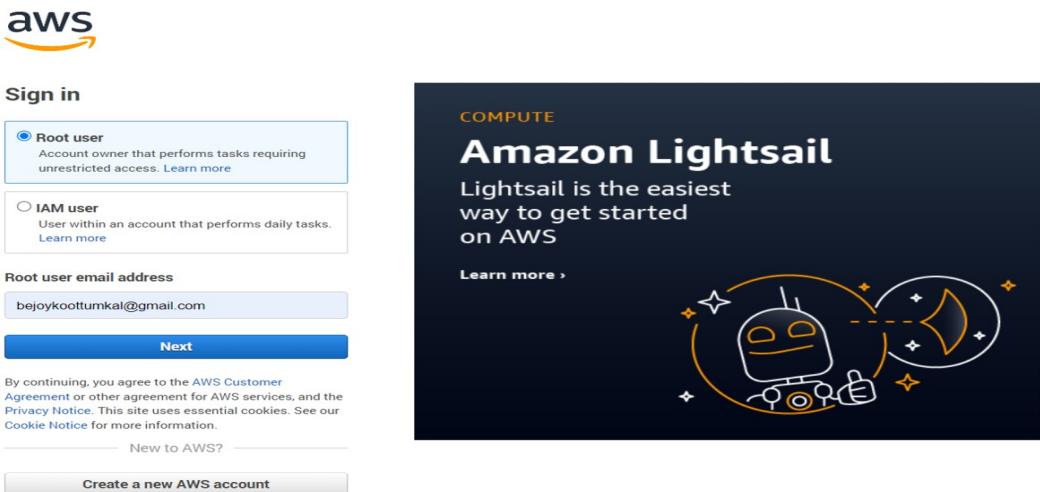
## **Task 4:**

**Integrate Docker host with Ansible. Write ansible playbook to create Image and create continuer. Integrate Ansible with Jenkins. Deploy ansible-playbook. CI/CD job to build code on ansible and deploy it on docker container**

- 1. Deploy Artifacts on Kubernetes**
- 2. Write pod, service, and deployment manifest file**
- 3. Integrate Kubernetes with ansible**
- 4. Ansible playbook to create deployment and service**

## Approach i have followed:

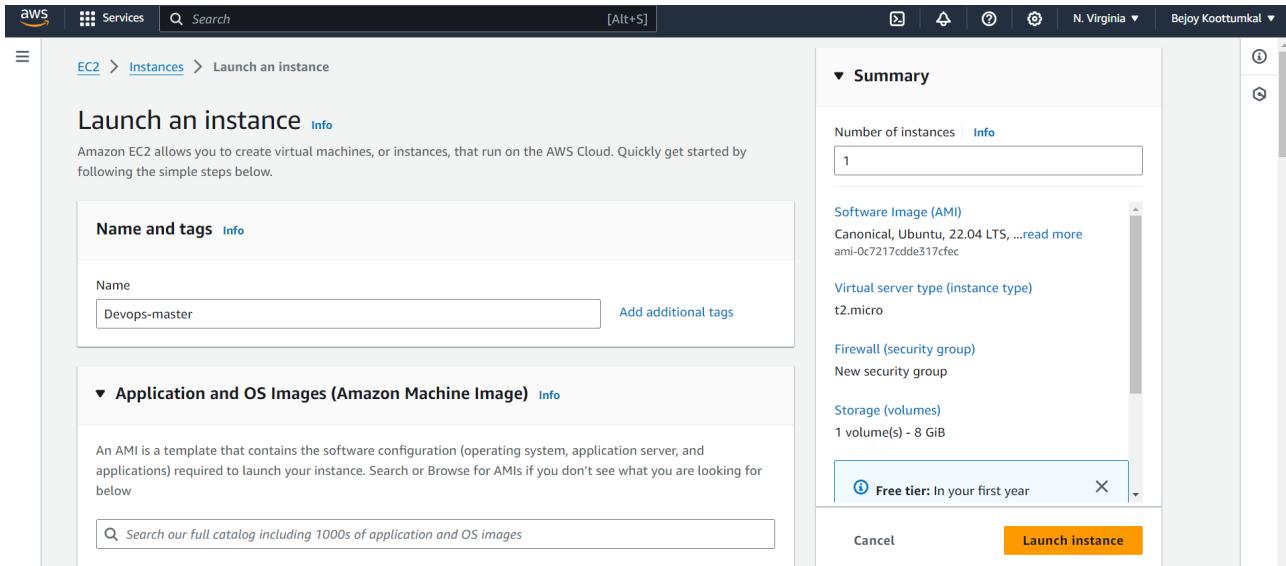
### Step1: Login to AWS



## EC2 Dashboard.

The image shows the EC2 Dashboard in the AWS Management Console. The left sidebar lists navigation options like EC2 Dashboard, EC2 Global View, Events, and Instances. The main area displays 'Resources' with counts for Instances (running), Auto Scaling Groups, Dedicated Hosts, Elastic IPs, Instances, Key pairs, Load balancers, Placement groups, Security groups, Snapshots, and Volumes. Below this are sections for 'Launch instance' (with 'Launch instance' and 'Migrate a server' buttons) and 'Service health' (linking to the AWS Health Dashboard). The 'Zones' section shows a table with columns for Zone name and Zone ID. On the right, there's a 'EC2 Free Tier' summary, a note about 2 EC2 free tier offers in use, and a 'Offer usage (monthly)' section showing Linux EC2 Instances at 15% usage and Storage space on EBS at 100% usage, with a note about an offer limit reached.

## Step 2: Created an EC2 Instance with t2 micro- devops-master.



**This instance was used for launching Kubernetes Master and Client nodes**

## Step 3: Set Up Master and Client Nodes:

I followed Kubernetes Installation using the following Link

### Reference:

<https://github.com/yankils/Simple-DevOps-Project/blob/master/Kubernetes>

[Kubernetes Setup using kops.md](#)

**After the installation, i verified the kubernetes master and client node status using the command: # kops validate cluster**

```
root@ip-172-31-18-184:~# kops validate cluster
Using cluster from kubectl context: demo.k8s.bejoy.net
Validating cluster demo.k8s.bejoy.net

INSTANCE GROUPS
NAME          ROLE      MACHINETYPE   MIN   MAX   SUBNETS
control-plane-us-east-1a  ControlPlane  t3.medium    1     1     us-east-1a
nodes-us-east-1a    Node       t3.medium    1     1     us-east-1a

NODE STATUS
NAME          ROLE      READY
l-0484700bce1f1399f  node      True
l-0d35f036c36cd42b5  control-plane  True

Your cluster demo.k8s.bejoy.net is ready
root@ip-172-31-18-184:~#
```

## Step 4: From AWS EC2 Console launched another machine for Ansible Configuration.

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Devops-master	i-0b0861cdd650cc495	Stopped	t2.micro	-	View alarms	us-east-1b	-
Ansible	i-0d31f3164dd9ef61f	Stopped	t2.micro	-	View alarms	us-east-1b	-

### Reference:

<https://www.cherryservers.com/blog/install-ansible-ubuntu>

After the installation created a work folder for Ansible: **/opt/k8s-lab**

Copied the files from : <https://github.com/bejoykoottumkal/devops-lab/tree/main/cicd/ansible> to the ansible work folder.

The screenshot shows three terminal windows:

- Terminal 1 (root):

```
root@ip-172-31-18-104:~
```
- Terminal 2 (ubuntu):

```
ubuntu@i-0f4294bfed30477c4:~
```
- Terminal 3 (ubuntu):

```
ubuntu@ip-172-31-17-80:/opt/k8s-lab
```

In Terminal 3, the command `ls` is run, showing the following files:  
create-simple-devops-image.yml Dockerfile hosts kubernetes-devops-lab-deployment.yml kubernetes-devops-lab-service.yml webapp.war

## Step 5: Copied the deploy and service files to K8s master.

SSH to K8s Master and copied the files from:

<https://github.com/bejoykoottumkal/devops-lab/tree/main/cicd/k8s>

To

/home/ubuntu

## Step 6: Integrated Ansible with Jenkins

The Ansible plugin was installed on Jenkins through the following steps:

Navigated to Jenkins Dashboard -> Manage Jenkins -> Manage Plugins -> Available.

Searched for "Ansible" and installed the plugin.

Ansible was configured in Jenkins:

The screenshot shows the Jenkins Manage Jenkins > Plugins page. A search bar at the top contains the text 'ansible'. Below it, a table lists two plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	Ansible 307.va_1f3ef06575a...	22 days ago
<input type="checkbox"/>	Ansible Tower 0.16.0 This plugin connects Jenkins with Ansible Tower	3 yr 7 mo ago

On the left sidebar, the 'Available plugins' tab is selected. Other tabs include 'Updates', 'Installed plugins', 'Advanced settings', and 'Download progress'.

The screenshot shows the Jenkins Manage Jenkins > Plugins page. The 'Download progress' tab is selected. It displays the progress of installing the Ansible plugin, showing various steps and their success status:

Preparation	
Cloud Statistics	<input checked="" type="checkbox"/> Success
Docker Commons	<input checked="" type="checkbox"/> Success
Apache HttpComponents Client 5.x API	<input checked="" type="checkbox"/> Success
Docker API	<input checked="" type="checkbox"/> Success
Docker	<input checked="" type="checkbox"/> Success
Loading plugin extensions	<input checked="" type="checkbox"/> Success
Ansible	<input checked="" type="checkbox"/> Success
Loading plugin extensions	<input checked="" type="checkbox"/> Success

Below the table, there are two links:

- [Go back to the top page](#)  
(you can start using the installed plugins right away)
- Restart Jenkins when installation is complete and no jobs are running

On the left sidebar, the 'Available plugins' tab is selected. Other tabs include 'Updates', 'Installed plugins', 'Advanced settings', and 'Download progress'.

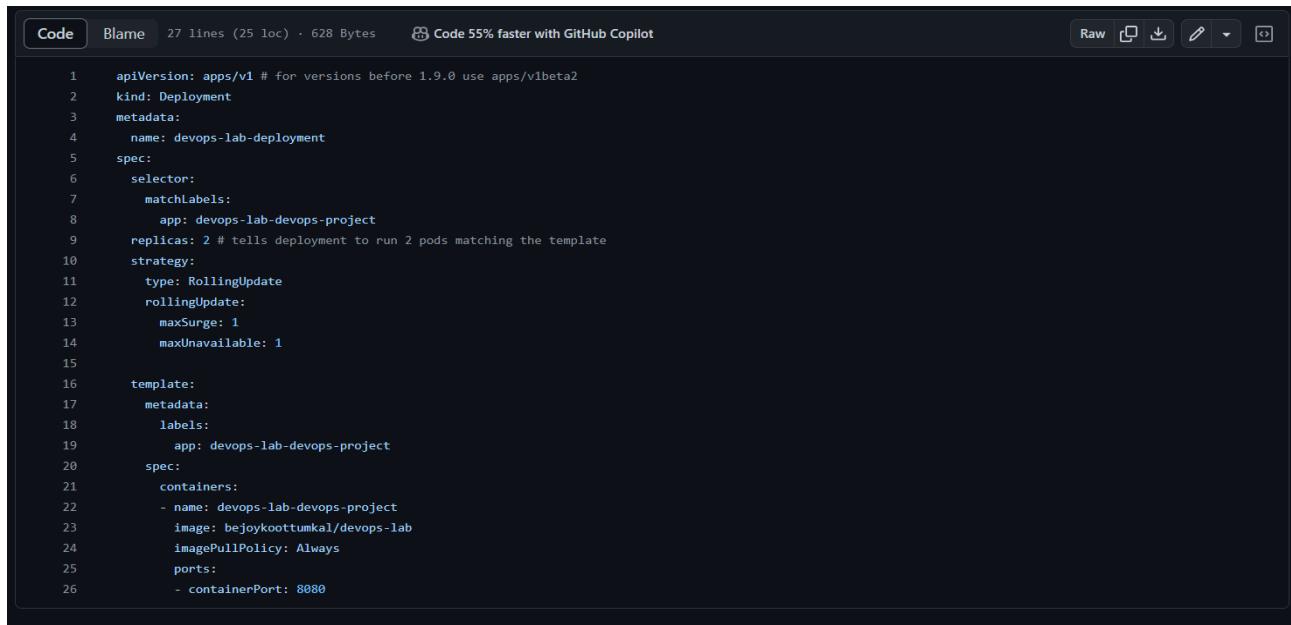
## Step 7: Deployed ansible-playbook through pipeline script.

```
stage('Deploy to K8s') {
    steps {
        sshagent(credentials: ['ssh_ansible']) {
            sh 'scp $WORKSPACE/webapp/target/webapp.war ubuntu@$ANSIBLE_SERVER_IP:/opt/k8s-lab/'
        }
        sshagent(credentials: ['ssh_ansible']) {
            sh 'ssh -o StrictHostKeyChecking=no -l ubuntu $ANSIBLE_SERVER_IP ansible-playbook -i /opt/k8s-lab/hosts /opt/k8s-lab/create-simple-devops-image.yml'
        }
        sshagent(credentials: ['ssh_ansible']) {
            sh 'ssh -o StrictHostKeyChecking=no -l ubuntu $ANSIBLE_SERVER_IP ansible-playbook -i /opt/k8s-lab/hosts /opt/k8s-lab/kubernetes-devops-lab-deployment.yml'
        }
        sshagent(credentials: ['ssh_ansible']) {
            sh 'ssh -o StrictHostKeyChecking=no -l ubuntu $ANSIBLE_SERVER_IP ansible-playbook -i ansible-playbook -i /opt/k8s-lab/hosts /opt/k8s-lab/kubernetes-devops-lab-service.yml'
        }
    }
}
```

## Step 8: Deployed Artifacts on Kubernetes

Kubernetes manifest files were written to define Deployment and Service configurations.

An Ansible playbook for Kubernetes deployment, cicd/k8s/devops-lab-deploy.yml, was created:



```
Code Blame 27 lines (25 loc) · 628 Bytes ⚡ Code 55% faster with GitHub Copilot Raw ⌂ ⌄ ⌅ ⌆ ⌇
```

```
1  apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
2  kind: Deployment
3  metadata:
4      name: devops-lab-deployment
5  spec:
6      selector:
7          matchLabels:
8              app: devops-lab-devops-project
9      replicas: 2 # tells deployment to run 2 pods matching the template
10     strategy:
11         type: RollingUpdate
12         rollingUpdate:
13             maxSurge: 1
14             maxUnavailable: 1
15
16     template:
17         metadata:
18             labels:
19                 app: devops-lab-devops-project
20         spec:
21             containers:
22                 - name: devops-lab-devops-project
23                     image: bejoykoottumkal/devops-lab
24                     imagePullPolicy: Always
25                 ports:
26                     - containerPort: 8080
```



```
1  apiVersion: v1
2  kind: Service
3  metadata:
4      name: devops-lab-service
5      labels:
6          app: devops-lab-devops-project
7  spec:
8      selector:
9          app: devops-lab-devops-project
10     type: LoadBalancer
11     ports:
12         - port: 8080
13             targetPort: 8080
14             nodePort: 31200
```

## Step 8: White listing Application Port in AWS Security group

### Enabling 31200 in AWS security group

The screenshot shows the AWS Security Groups console. A new rule is being added at the bottom:

Source	Protocol	Port Range	Action	Target Group	Tags	Actions
sg-07971e58ca37fefef	TCP	1 - 2379	Custom			Delete
sg-07971e58ca37fefef	TCP	443	Custom			Delete
sg-07971e58ca37fefef	TCP	2382 - 4000	Custom			Delete
sg-07971e58ca37fefef	TCP	4003 - 6553	Custom			Delete
sg-07971e58ca37fefef	UDP	1 - 65535	Custom			Delete
-	TCP	31200	Anyw...	0.0.0.0/0	sg-07971e58ca37fefef	Add rule

## Step 9: Running the CICD Pipeline by doing a Test Commit and Pushed to master branch.

The screenshot shows the Jenkins CI\_PIPELINE pipeline stage view. Three builds are listed:

Build	Time
#137 Jan 18, 2024, 1:25 PM	139ms
#136 Jan 18, 2024, 11:28 AM	125ms
#135	123ms

Each build row includes a timeline bar showing the duration of each stage:

Stage	Time
Declarative: Tool Install	120ms
Stage 1 - Checkout Code	1s
Stage 2 - Compile Code	1s
Stage 3 - Run Unit Tests	1s
Stage 4 - Create package	1s
Stage 5 - clean install	2s
Build Docker Image	3s
Login to Docker Hub	3s
Push Image to Docker Hub	8s
Deploy to K8s	10s
Declarative: Post Actions	48ms

## Console Output:

Dashboard > CI\_PIPELINE > #128

```
PLAY RECAP ****
54.145.170.236 : ok=2     changed=1      unreachable=0    failed=0     skipped=0    rescued=0    ignored=0

[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 39969 killed;
[ssh-agent] Stopped.
[Pipeline] // sshagent
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Dashboard > CI\_PIPELINE > #128

```
(/var/lib/jenkins/workspace/CI_PIPELINE@tmp/private_key_18166783774982839877.key)
[ssh-agent] Started.
[Pipeline] {
[Pipeline] sh
+ ssh -o StrictHostKeyChecking=no -l ubuntu 52.90.96.184 ansible-playbook -i ansible-playbook -i /opt/k8s-lab/opt/k8s-lab/kubernetes-devops-lab-service.yml
[WARNING]: Unable to parse /home/ubuntu/ansible-playbook as an inventory source
[WARNING]: Invalid characters were found in group names but not replaced, use
-vvvv to see details

PLAY [create service for deployment] *****

TASK [Gathering Facts] *****
ok: [54.145.170.236]

TASK [create a service] *****
changed: [54.145.170.236]

PLAY RECAP ****
54.145.170.236 : ok=2     changed=1      unreachable=0    failed=0     skipped=0    rescued=0    ignored=0

[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 39969 killed;
```

Dashboard > CI\_PIPELINE > #128

```
[Pipeline] sh
+ sudo docker build -t bejoykoottumkal/devops-lab:128 .
#0 building with "default" instance using docker driver

#1 [internal] load .dockerignore
#1 transferring context: 2B done
#1 DONE 0.0s

#2 [internal] load build definition from Dockerfile
#2 transferring dockerfile: 169B done
#2 DONE 0.0s

#3 [internal] load metadata for docker.io/library/tomcat:latest
#3 ...

#4 [auth] library/tomcat:pull token for registry-1.docker.io
#4 DONE 0.0s

#3 [internal] load metadata for docker.io/library/tomcat:latest
#3 DONE 2.8s

#5 [1/3] FROM docker.io/library/tomcat:latest@sha256:504823cf27f257059521ba0b9d40e88761109fd78f6b7bc3212747346
#5 DONE 0.0s

#6 [internal] load build context
#6 transferring context: 2B 0.0s done
```

Dashboard > CI\_PIPELINE > #128

```
-----
T E S T S
-----

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] Reactor Summary for Maven Project 1.0-SNAPSHOT:
[INFO]
[INFO] Maven Project ..... SUCCESS [ 0.001 s]
[INFO] Server ..... SUCCESS [ 0.486 s]
[INFO] Webapp ..... SUCCESS [ 0.088 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  0.629 s
[INFO] Finished at: 2024-01-16T18:29:13+05:30
[INFO] -----
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
```

Dashboard > CI\_PIPELINE > #128

```

[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] -----< com.example.maven-project:webapp >-----
[INFO] Building Webapp 1.0-SNAPSHOT [3/3]
[INFO] -----[ war ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.5:resources (default-resources) @ webapp ---
[debug] execute contextualize
[INFO] Using 'utf-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /var/lib/jenkins/workspace/CI_PIPELINE/webapp/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:compile (default-compile) @ webapp ---
[INFO] No sources to compile
[INFO]
[INFO] -----
[INFO] Reactor Summary for Maven Project 1.0-SNAPSHOT:
[INFO]
[INFO] Maven Project ..... SUCCESS [ 0.001 s]
[INFO] Server ..... SUCCESS [ 0.246 s]
[INFO] Webapp ..... SUCCESS [ 0.010 s]
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO]
[INFO] -----
[INFO] Total time:  0.311 s
[INFO] Finished at: 2024-01-16T18:29:12+05:30
[INFO]
[INFO] -----
[Pipeline] 

```

## AWS Instances Up and Running.

The screenshot shows the AWS CloudWatch Metrics Insights interface. On the left, there's a sidebar with navigation links like 'New', 'Images', 'AMI Catalog', 'Elastic Block Store', 'Volumes', 'Snapshots', 'Lifecycle Manager', 'Network & Security', 'Security Groups', 'Elastic IPs', 'Placement Groups', 'Key Pairs', 'Network Interfaces', 'Load Balancing', 'Load Balancers', 'Target Groups', 'Trust Stores New', and 'Auto Scaling' with 'Auto Scaling Groups'. The main area is titled 'Instances (4) Info' and contains a table with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability. The table lists four instances: 'nodes-us-east-1a.demo.k8s.bejoy.net' (running, t3.medium, 2/2 checks passed, View alarms, us-east), 'control-plane-us-east-1a.masters.demo.k8s.be...' (running, t3.medium, 2/2 checks passed, View alarms, us-east), 'Devops-master' (running, t2.micro, 2/2 checks passed, View alarms, us-east), and 'Ansible' (running, t2.micro, 2/2 checks passed, View alarms, us-east). Below the table, there's a section titled 'Select an instance'.

## Application Live Screenshot:

Application up and running in <http://54.166.216.219:31200/webapp/>



This is retail portal developed By Bejoy - V2.7

[Add Product](#) [View Product](#)

## References:

[https://docs.ansible.com/ansible/latest/collections/community/docker/docker\\_image\\_module.Html](https://docs.ansible.com/ansible/latest/collections/community/docker/docker_image_module.Html)  
[https://docs.ansible.com/ansible/latest/collections/community/docker/docker\\_container\\_module.html](https://docs.ansible.com/ansible/latest/collections/community/docker/docker_container_module.html)  
<https://github.com/ansible-collections/community.docker>  
<https://pypi.org/project/docker/>  
<https://pypi.org/project/docker-py/>  
<https://github.com/ansible-collections/community.docker>  
<https://docker-py.readthedocs.io/en/latest/>  
<https://www.baeldung.com/ops/docker-removing-images#:~:text=Forcefully%20Remove%20Containers%20and%20Images&text=The%20%2Df%20flag%20is%20used,the%20running%20Docker%20containers%20forcefully.&text=The%20docker%20images%20%2Dqa%20will,forcefully%20remove%20the%20Docker%20image>  
<https://stackoverflow.com/questions/22769568/system-specific-variables-in-ansible>  
<https://stackoverflow.com/questions/60834692/how-to-completely-remove-ansible-2-8-3-on-ubuntu-18-04-->  
[https://docs.ansible.com/ansible/2.6/modules/k8s\\_module.html#examples](https://docs.ansible.com/ansible/2.6/modules/k8s_module.html#examples)  
[https://docs.ansible.com/ansible/latest/user\\_guide/guide\\_rolling\\_upgrade.html](https://docs.ansible.com/ansible/latest/user_guide/guide_rolling_upgrade.html)  
<https://adamtheautomator.com/ansible-kubernetes/>  
[https://adamtheautomator.com/ansible-template/ansible-playbook%20sample-playbook.yml%20-e%20'ansible\\_python\\_interpreter=/usr/bin/python3'%20-](https://adamtheautomator.com/ansible-template/ansible-playbook%20sample-playbook.yml%20-e%20'ansible_python_interpreter=/usr/bin/python3'%20-)  
[https://docs.ansible.com/ansible/latest/reference\\_appendices/python\\_3\\_support.html](https://docs.ansible.com/ansible/latest/reference_appendices/python_3_support.html)  
[https://docs.ansible.com/ansible/2.9/modules/k8s\\_service\\_module.html](https://docs.ansible.com/ansible/2.9/modules/k8s_service_module.html)  
<https://www.youtube.com/watch?v=NSk0NHkTjDs>  
<https://www.baeldung.com/ops/jenkins-parameterized-builds>  
<https://stackoverflow.com/questions/56003777/how-to-pass-environment-variable-in-kubectl#deployment>  
<https://stackoverflow.com/questions/62108860/kubectl-no-matches-for-kind-service-in-version-apps-v1>  
<https://stackoverflow.com/questions/59579482/kubectl-apply-error-from-server-forbidden-authentication-required-jenkins>  
<https://stackoverflow.com/questions/51489359/docker-using-password-via-the-cli-is-insecure-use-password-stdin>  
<https://kubernetes.io/docs/reference/kubectl/quick-reference/>

## Task 5:

Using Prometheus monitor the resources like CPU utilization: Total Usage, Usage per core, usage breakdown, Memory , Network on the instance by providing the end points in local host. Install node exporter and add URL to target in Prometheus. Using this data login to Grafana and create a dashboard to show the metrics

Approach i have followed:

### Step 1: Installed Prometheus

- SSH to Kubernetes Master
- Installed Prometheus
- Find the port listening for prometheus
- Prometheus-server-ext => 31091

```
https://prometheus.io/
ubuntu@i-0f4294bfed38477c4: $ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
devops-lab-deployment-6cd44464c8-498qm   1/1     Running   0          2m44s
devops-lab-deployment-6cd44464c8-89jxc   1/1     Running   0          2m44s
prometheus-alertmanager-0                1/1     Running   0          25s
prometheus-kube-state-metrics-6b464f5b88-wpqws   1/1     Running   0          25s
prometheus-prometheus-node-exporter-q56xq   1/1     Running   0          25s
prometheus-prometheus-node-exporter-x69ns   1/1     Running   0          25s
prometheus-prometheus-pushgateway-7857c44f49-zwj9x  1/1     Running   0          25s
prometheus-server-6b68fb5d54b-swrjn   1/2     Running   0          25s

ubuntu@i-0f4294bfed38477c4: $ kubectl get svc
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
devops-lab-service   LoadBalancer   100.71.43.123 <pending>   8080:31200/TCP   2m49s
kubernetes       ClusterIP    100.64.0.1    <none>        443/TCP    34m
prometheus-alertmanager   ClusterIP    100.67.239.61 <none>        9093/TCP   37s
prometheus-alertmanager-headless   ClusterIP    None         <none>        9093/TCP   37s
prometheus-kube-state-metrics   ClusterIP    100.71.217.38 <none>        8080/TCP   37s
prometheus-prometheus-node-exporter   ClusterIP    100.67.252.73 <none>        9100/TCP   37s
prometheus-prometheus-pushgateway   ClusterIP    100.71.186.44 <none>        9091/TCP   37s
prometheus-server       ClusterIP    100.66.206.103 <none>        80/TCP     37s

ubuntu@i-0f4294bfed38477c4: $ kubectl expose service prometheus-server --type=NodePort --target-port=9090 --name=prometheus-server-ext
service/prometheus-server-ext exposed
ubuntu@i-0f4294bfed38477c4: $ 
```

```
root@ip-172-31-18-184:~# kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
devops-lab-deployment-8674699c99-g9bzd   1/1     Running   0          6m11s
devops-lab-deployment-8674699c99-x2xbv   1/1     Running   0          6m11s
prometheus-alertmanager-0                0/1     ContainerCreating   0          99m
prometheus-kube-state-metrics-6b464f5b88-q7hm4   1/1     Running   0          99m
prometheus-prometheus-node-exporter-f2kft   1/1     Running   0          98m
prometheus-prometheus-node-exporter-wd249   1/1     Running   0          101m
prometheus-prometheus-pushgateway-7857c44f49-2gz8w  1/1     Running   0          99m
prometheus-server-6b68fb5d54b-bdz72   0/2     ContainerCreating   0          99m
root@ip-172-31-18-184:~# kubectl logs -t tail=50 devops-lab-deployment-8674699c99-g9bzd
18-Jan-2024 05:59:24.743 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version name: Apache Tomcat/10.1.18
18-Jan-2024 05:59:24.759 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built: Jan 5 2024 14:39:40 UTC
18-Jan-2024 05:59:24.759 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version number: 10.1.18.0
18-Jan-2024 05:59:24.759 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name: Linux
18-Jan-2024 05:59:24.759 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Version: 6.2.0-1017-aws
18-Jan-2024 05:59:24.760 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Architecture: amd64
18-Jan-2024 05:59:24.760 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home: /opt/java/openjdk
18-Jan-2024 05:59:24.762 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Version: 21.0.1+12-LTS
18-Jan-2024 05:59:24.762 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log JVM Vendor: Eclipse Adoptium
18-Jan-2024 05:59:24.762 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_BASE: /usr/local/tomcat
18-Jan-2024 05:59:24.763 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log CATALINA_HOME: /usr/local/tomcat
18-Jan-2024 05:59:24.797 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.config.file=/usr/local/tomcat/conf/logging.properties
18-Jan-2024 05:59:24.808 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
18-Jan-2024 05:59:24.820 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djdk.tls.ephemeralDHKeySize=2048
18-Jan-2024 05:59:24.821 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Djava.protocol.handler.pkgs=org.apache.catalina.webresources
18-Jan-2024 05:59:24.821 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: -Dorg.apache.catalina.security.SecurityListener.UMASK=0027
18-Jan-2024 05:59:24.821 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: --add-opens=java.base/java.lang=ALL-UNNAMED
18-Jan-2024 05:59:24.821 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: --add-opens=java.base/java.io=ALL-UNNAMED
18-Jan-2024 05:59:24.821 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: --add-opens=java.base/java.util.concurrent=ALL-UNNAMED
18-Jan-2024 05:59:24.821 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: --add-opens=java.net/java.net=ALL-UNNAMED
18-Jan-2024 05:59:24.822 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: --add-opens=javalin.base=/usr/local/tomcat
18-Jan-2024 05:59:24.822 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Command line argument: --add-opens=javalin.home=/usr/local/tomcat
18-Jan-2024 05:59:24.845 INFO [main] org.apache.catalina.core.AprLifecycleListener.lifecycleEvent Loaded Apache Tomcat Native library [2.0.6] using APR version [1.7.0].
18-Jan-2024 05:59:24.855 INFO [main] org.apache.catalina.core.AprLifecycleListener.initializeSSL OpenSSL successfully initialized [OpenSSL 3.0.2 15 Mar 2022]
18-Jan-2024 05:59:26.015 INFO [main] org.apache.coyote.AbstractProtocol.init Initializing ProtocolHandler ["http-nio-8080"]
18-Jan-2024 05:59:26.082 INFO [main] org.apache.catalina.startup.Catalina.load Server initialization in [2224] milliseconds
18-Jan-2024 05:59:26.217 INFO [main] org.apache.catalina.core.StandardService.startInternal Starting service [Catalina]
18-Jan-2024 05:59:26.218 INFO [main] org.apache.catalina.core.StandardEngine.startInternal Starting Servlet engine: [Apache Tomcat/10.1.18]
18-Jan-2024 05:59:26.308 INFO [main] org.apache.catalina.startup.HostConfig.deployWAR Deploying web application archive [/usr/local/tomcat/webapps/webapp.war]
18-Jan-2024 05:59:27.173 INFO [main] org.apache.catalina.startup.HostConfig.deployWAR Deployment of web application archive [/usr/local/tomcat/webapps/webapp.war] has finished in [866] ms
18-Jan-2024 05:59:27.183 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8080"]
18-Jan-2024 05:59:27.201 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in [1112] milliseconds
containing 172 lines. # ]
```

```
ubuntu@i-0f4294bfed38477c4:~$ kubectl get svc
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP  PORT(S)        AGE
devops-lab-service LoadBalancer  100.71.43.123 <pending>  8080:31200/TCP  8m30s
kubernetes       ClusterIP  100.64.0.1    <none>       443/TCP       39m
prometheus-alertmanager ClusterIP  100.67.239.61 <none>       9093/TCP     6m18s
prometheus-alertmanager-headless ClusterIP  None        <none>       9093/TCP     6m18s
prometheus-kube-state-metrics ClusterIP  100.71.217.38 <none>       8080/TCP     6m18s
prometheus-prometheus-node-exporter ClusterIP  100.67.252.73 <none>       9100/TCP     6m18s
prometheus-prometheus-pushgateway ClusterIP  100.71.186.44 <none>       9091/TCP     6m18s
prometheus-server ClusterIP  100.66.206.103 <none>       80/TCP       6m18s
prometheus-server-ext NodePort   100.66.206.103 <none>       80/TCP       6m18s
ubuntu@i-0f4294bfed38477c4:~$
```

## Step 2 :White listing Prometheus Port in AWS Security group

- Open port 31091 in AWS security group

Stack Name	Output Key	Type	Protocol	Port Range	Custom VPC	Target Security Group	Action
sgr-070aa4061c63174f6	Custom TCP	TCP	1 - 2379		Custom	sg-07971e58ca37efef	Delete
sgr-01dc50ceafdf5b1a29	HTTPS	TCP	443		Custom	:/0	Delete
sgr-0455bf4660ffe792a	Custom TCP	TCP	2382 - 4000		Custom	sg-07971e58ca37efef	Delete
sgr-0549cea37ff112100	Custom TCP	TCP	4003 - 6555		Custom	sg-07971e58ca37efef	Delete
sgr-01269dc49414e39c0	Custom UDP	UDP	1 - 65535		Custom	sg-07971e58ca37efef	Delete
-	Custom TCP	TCP	31200	Anyw...	Custom	0.0.0.0/0	0.0.0.0/0

[Add rule](#)

## Step 3: Prometheus Monitoring

Prometheus started at url - <http://44.203.157.199:31091>

The screenshot shows the Prometheus web interface with a single panel. The URL in the address bar is <http://44.203.157.199:31091/graph>. The interface includes a top navigation bar with links for Prometheus, Alerts, Graph, Status, Help, and a user icon. Below the navigation is a search bar with the placeholder "Expression (press Shift+Enter for newlines)". Underneath the search bar are two tabs: "Table" and "Graph". A message "No data queried yet" is displayed. At the bottom right of the panel is a "Remove Panel" button.

## Service Discovery

The screenshot shows the Prometheus Service Discovery interface. At the top, there are three tabs: 'bejoykoottumka/devops' (active), '44.203.157.199:31200/web' (inactive), and 'Prometheus Time Series' (inactive). Below the tabs, the URL is 44.203.157.199:31091/service-discovery?search=#kubernetes-nodes. The main content area is titled 'Service Discovery' and contains a search bar and a 'Filter by labels' dropdown. A list of discovered targets is shown:

- kubernetes-apiservers (1 / 17 active targets)
- kubernetes-nodes (2 / 2 active targets)
- kubernetes-nodes-cadvisor (2 / 2 active targets)
- kubernetes-service-endpoints (5 / 13 active targets)
- prometheus (1 / 1 active targets)
- prometheus-pushgateway (1 / 12 active targets)

Below this, a specific target is expanded: 'kubernetes-apiservers'. It shows 'Discovered Labels' and a long list of labels for the target, including:  
address = "172.20.205.9:443"  
meta\_kubernetes\_endpoint\_port\_name = "https"  
meta\_kubernetes\_endpoint\_port\_protocol = "TCP"  
meta\_kubernetes\_endpoint\_ready = "true"  
meta\_kubernetes\_endpoint\_label\_endpointslice\_kubernetes\_to\_skip\_mirror = "true"  
meta\_kubernetes\_endpoint\_labelpresent\_endpointslice\_kubernetes\_to\_skip\_mirror = "true"  
meta\_kubernetes\_endpoints\_name = "kubernetes"  
meta\_kubernetes\_namespace = "default"  
meta\_kubernetes\_service\_label\_component = "apiserver"  
meta\_kubernetes\_service\_label\_provider = "kubernetes"  
meta\_kubernetes\_service\_labelpresent\_component = "true"  
meta\_kubernetes\_service\_labelpresent\_provider = "true"  
meta\_kubernetes\_service\_name = "kubernetes"  
metrics\_path = "/metrics"  
scheme = "https"  
scrape\_interval = "1m"  
scrape\_timeout = "10s"  
job = "kubernetes-apiservers"  
  
address = "108.94.1.248:9090"  
meta\_kubernetes\_endpoint\_address\_target\_kind = "Pod"  
meta\_kubernetes\_endpoint\_address\_target\_name = "prometheus-server-4b6fbfd54b-swjn"  
meta\_kubernetes\_endpoint\_node\_name = "1-08c1d005e0e52698e"

## Configuration

The screenshot shows the Prometheus Configuration interface. At the top, there are three tabs: 'bejoykoottumka/devops' (active), '44.203.157.199:31200/web' (inactive), and 'Prometheus Time Series' (inactive). Below the tabs, the URL is 44.203.157.199:31091/config. The main content area is titled 'Configuration' and contains a 'Copy to clipboard' button. The configuration file content is displayed as follows:

```
global:  
  scrape_interval: 1m  
  scrape_timeout: 10s  
  evaluation_interval: 1m  
alerting:  
  alertmanagers:  
    - authorization:  
        type: Bearer  
        credentials_file: /var/run/secrets/kubernetes.io/serviceaccount/token  
    tls_config:  
      ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt  
      insecure_skip_verify: false  
      follow_redirects: true  
      enable_http2: true  
      scheme: http  
      timeout: 10s  
      api_version: v2  
      relabel_configs:  
        - source_labels: [__meta_kubernetes_namespace]  
          separator: ;  
          regex: default  
          replacement: $1  
          action: keep  
        - source_labels: [__meta_kubernetes_pod_label_app_kubernetes_io_instance]  
          separator: ;  
          regex: prometheus  
          replacement: $1  
          action: keep  
        - source_labels: [__meta_kubernetes_pod_label_app_kubernetes_io_name]  
          separator: ;  
          regex: alertmanager  
          replacement: $1  
          action: keep  
        - source_labels: [__meta_kubernetes_pod_container_port_number]  
          separator: ;  
          regex: "9093"  
          replacement: $1  
          action: keep  
  kubernetes_sd_configs:  
    - role: pod  
      kubernetes_sd_configs: ""  
      follow_redirects: true  
      enable_http2: true
```

## TSDB Status

Screenshot of the Prometheus TSDB Status page:

The page shows the following sections:

- Head Stats**: A table showing the number of series, chunks, label pairs, and time ranges.
- Head Cardinality Stats**: A table showing the top 10 label names with their value counts.
- Top 10 label names with value count**: A table listing label names and their counts.
- Top 10 series count by metric names**: A table listing metric names and their counts.

Number of Series	Number of Chunks	Number of Label Pairs	Current Min Time	Current Max Time
37648	74801	3390	2024-01-18T07:59:02.341Z (1705564742341)	2024-01-18T08:15:02.341Z (1705565702341)

Name	Count
__name__	1101
name	296
le	235
type	132
resource	104
id	94
mountpoint	59
kind	54
handler	53
resource_prefix	53

Name	Count
apiserver_request_duration_seconds_bucket	5904
etcd_request_duration_seconds_bucket	5064
...	...

## Runtime Information

Screenshot of the Prometheus Runtime Information page:

The page shows the following sections:

- Runtime Information**: A table listing various runtime parameters.
- Build Information**: A table listing build details.
- Alertmanagers**: A table listing alertmanager endpoints.

Start time	Thu, 18 Jan 2024 07:58:53 GMT
Working directory	/prometheus
Configuration reload	Successful
Last successful configuration reload	2024-01-18T08:01:49Z
WAL corruptions	0
Goroutines	141
GOMAXPROCS	2
GOMEMLIMIT	9223372036854776000
GOGC	
GODEBUG	
Storage retention	15d

Version	2.48.1
Revision	63894216648f0d6be310c9d16fb48293c45c9310
Branch	HEAD
BuildUser	root@71f108#5632
BuildDate	20231208-23:33:22
GoVersion	go1.21.5

Endpoint
<a href="http://100.96.1.91:9093/api/v2/alerts">http://100.96.1.91:9093/api/v2/alerts</a>

## Prometheus Targets

The screenshot shows the Prometheus Targets page with two tables of endpoint status. The first table, 'kubernetes-apiservers (1/1 up)', has one entry: https://172.20.205.9/metrics, which is UP with labels instance="172.20.205.9:443", job="kubernetes-apiservers", last scraped 39.737s ago, and a scrape duration of 162.492ms. The second table, 'kubernetes-nodes (2/2 up)', has three entries: kubernetes-nodes (UP), kubernetes-nodes-cadvisor (UP), and kubernetes-service-endpoints (5/5 up). The third table, 'prometheus (1/1 up)', has one entry: http://prometheus-prometheus-pushgateway.default.svc:9091/metrics, which is UP with labels instance="prometheus-prometheus-pushgateway.default.svc:9091", job="prometheus-pushgateway", last scraped 30.120s ago, and a scrape duration of 2.063ms.

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
https://172.20.205.9/metrics	UP	instance="172.20.205.9:443"; job="kubernetes-apiservers"	39.737s ago	162.492ms	

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
kubernetes-nodes	UP	instance=""; job="kubernetes-nodes"	30.120s ago	2.063ms	
kubernetes-nodes-cadvisor	UP	instance=""; job="kubernetes-nodes-cadvisor"			
kubernetes-service-endpoints	UP	instance=""; job="kubernetes-service-endpoints"			

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://prometheus-prometheus-pushgateway.default.svc:9091/metrics	UP	instance="prometheus-prometheus-pushgateway.default.svc:9091"; job="prometheus-pushgateway"	30.120s ago	2.063ms	

## Step 4: Monitored the Total Usage, Usage per core, usage breakdown, Memory, and Network on the instance Using the Prometheus

### 1) Total Usage:

Metric used: node\_cpu\_seconds\_total

Description: This metric represented the total CPU time consumed by the system.

PromQL Query:

```
sum(rate(node_cpu_seconds_total{mode="user"}[1m])) by (instance)
```

This PromQL query calculated the per-instance total CPU usage per second over the last 1 minute.

### 2) Usage per Core:

Metric used: node\_cpu\_seconds\_total

Description: CPU usage was broken down per core by using labels.

PromQL Query:

```
rate(node_cpu_seconds_total{mode="user"}[1m]) by (instance, cpu)
```

This query calculated the per-instance and per-core CPU usage per second over the last 1 minute.

### **3) Usage Breakdown:**

Metric used: node\_memory\_MemTotal, node\_memory\_MemFree, node\_memory\_Buffers, node\_memory\_Cached

Description: A breakdown of memory usage was obtained using various metrics related to memory.

PromQL Query:

```
node_memory_MemTotal - (node_memory_MemFree + node_memory_Buffers + node_memory_Cached)
```

This query calculated the used memory by subtracting free memory, buffer, and cache from the total memory.

### **4) Memory:**

Metric used: node\_memory\_MemTotal, node\_memory\_MemFree

Description: Total memory and free memory were monitored individually.

PromQL Queries:

```
node_memory_MemTotal
```

This query provided the total memory.

```
node_memory_MemFree
```

This query provided the amount of free memory.

### **5) Network:**

Metric used: node\_network\_receive\_bytes\_total, node\_network\_transmit\_bytes\_total

Description: Network usage was monitored.

PromQL Queries:

```
rate(node_network_receive_bytes_total[1m]) by (instance, device)
```

This query calculated the per-instance and per-network device incoming data rate over the last 1 minute.

```
rate(node_network_transmit_bytes_total[1m]) by (instance, device)
```

This query calculated the per-instance and per-network device outgoing data rate over the last 1 minute.

---

**THANK YOU**